1. Write a menu-driven python program to implement a stack of marks using user defined function :

a) PUSH() to push marks of a student in a stack.
b) POP() an element from the stack.
c) Display the stack of marks.

```
stack = []
top = -1


def push():
    global top
    elem = int(input("Enter Marks: "))
    stack.append(elem)
    top += 1
    print("PUSHED →", elem)


def pop():
    global top
    if not stack:
        print("Stack underflow (stack is empty)")
    else:
        elem = stack.pop()
        top -= 1
        print("POPPED ←", elem)


def display():
    global top
    if not stack:
        print("Stack underflow (stack is emtpy)")
    else:
        print("--------STACK START----------")
        print(stack[top], '← top')
        for i in range(top - 1, -1, -1):
            print(stack[i])
        print("--------STACK  END----------")


while True:
    choice = input(
        "1. Push\n"
        "2. Pop\n"
        "3. Display\n"
        "4. Exit\n"
        "\nYour Choice: "
    )
    if choice == "1":
        push()
        print('TOP: ', top)
    elif choice == "2":
        pop()
        print('TOP: ', top)
    elif choice == "3":
        display()
    elif choice == "4":
        break
    else:
        print("Invalid Choice…")
```

OUTPUT

```
1. Push
2. Pop
3. Display
4. Exit
```

```
Your Choice: 1
Enter Marks: 10
PUSHED -> 10
TOP:  0
1. Push
2. Pop
3. Display
4. Exit

Your Choice: 1
Enter Marks: 20
PUSHED -> 20
TOP:  1
1. Push
2. Pop
3. Display
4. Exit

Your Choice: 3
-------STACK START----------
20 <- top
10
-------STACK  END----------
1. Push
2. Pop
3. Display
4. Exit

Your Choice: 2
POPPED <- 20
TOP:  0
1. Push
2. Pop
3. Display
4. Exit

Your Choice: 3
-------STACK START----------
10 <- top
-------STACK  END----------
1. Push
2. Pop
3. Display
4. Exit

Your Choice: 4
```

2. WAP to reverse a string using PUSH and POP operation in Stack.

```
stack = []
top = -1


def push(elem):
    global top
    stack.append(elem)
    top += 1
    print("PUSHED ->", elem)


def pop():
    global top
    if not stack:
        print("Stack underflow (stack is empty)")
    else:
        elem = stack.pop()
        top -= 1
        print("POPPED <-", elem)
        return elem


def reverse_string():
    string = input("String To Reverse: ")
    [push(char) for char in string]

    reverse = ""
    for i in range(len(string)):
        reverse += pop()

    print("OUTPUT: ", reverse)


reverse_string()
```

OUTPUT

```
String To Reverse: hell
PUSHED -> h
PUSHED -> e
PUSHED -> l
PUSHED -> l
POPPED <- l
POPPED <- l
POPPED <- e
POPPED <- h
OUTPUT:  lleh
```

3. Write a program to create a Stack for storing only odd numbers out of all the numbers entered by the user.
Display the content of the Stack along with the largest odd number in the Stack.
(Hint : Keep popping out the elements from stack and maintain the largest element retrieved so far in a variable.
Repeat till Stack is empty)

```
stack = []
top = -1


def push(elem):
    global top
    stack.append(elem)
    top += 1
    print("PUSHED ->", elem)


def pop():
    global top
    if not stack:
        print("Stack underflow (stack is empty)")
    else:
        elem = stack.pop()
        top -= 1
        print("POPPED <-", elem)
        return elem


def display():
    global top
    if not stack:
        print("Stack underflow (stack is emtpy)")
    else:
        print("-------STACK START----------")
        print(stack[top], '<- top')
        for i in range(top - 1, -1, -1):
            print(stack[i])
        print("-------STACK  END----------")


def odd_input():
    largest_elem = None
    while input("Enter numbers (y/n): ").lower() == "y":
        num = int(input("Enter number: "))
        if largest_elem is None:
            largest_elem = num
        if num % 2 != 0:
            push(num)
            largest_elem = max(num, largest_elem)


odd_input()
print("Stack: ")
display()
```

OUTPUT

```
Enter numbers (y/n): y
Enter number: 1
PUSHED -> 1
Enter numbers (y/n): y
Enter number: 2
Enter numbers (y/n): y
Enter number: 3
PUSHED -> 3
Enter numbers (y/n): y
Enter number: 4
Enter numbers (y/n): y
Enter number: 5
PUSHED -> 5
Enter numbers (y/n): n
Stack:
-------STACK START----------
5 <- top
3
1
-------STACK  END----------
```